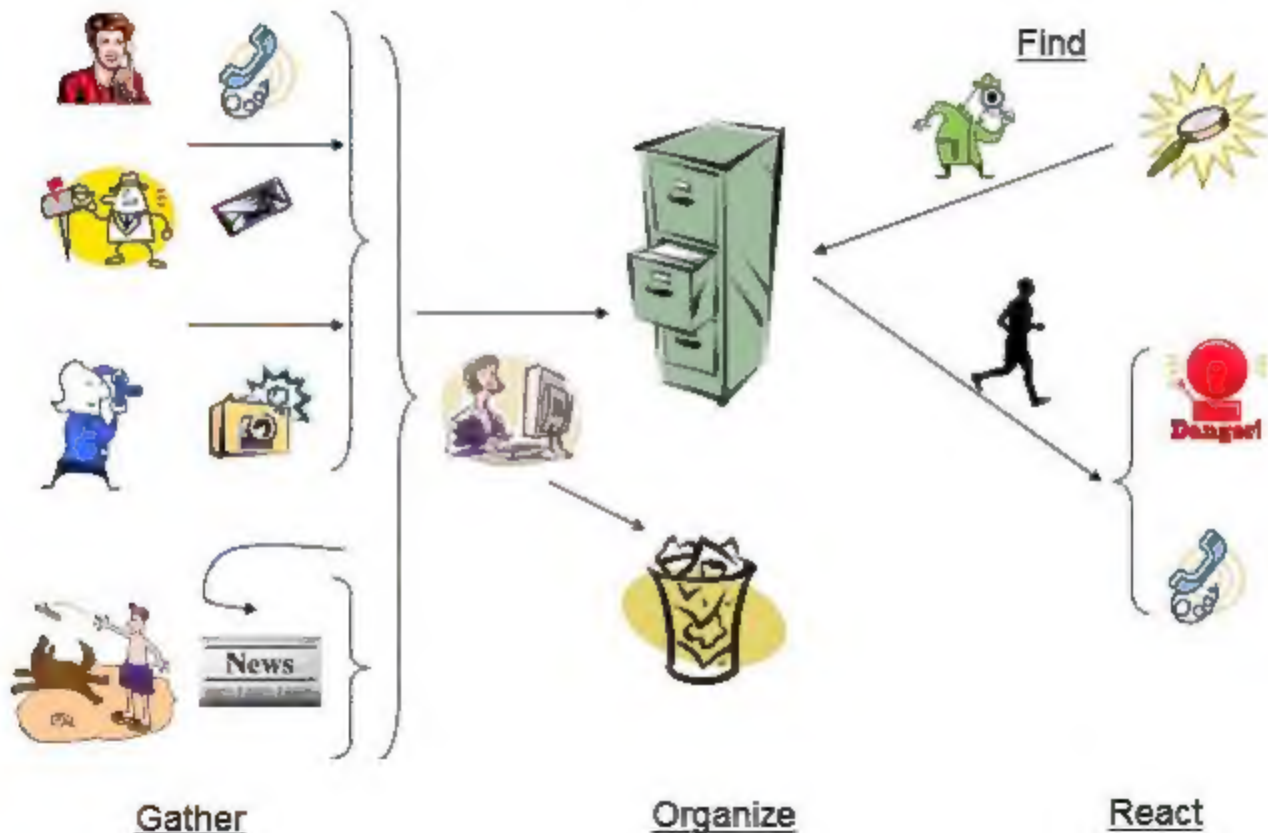


WinFS Rules

Praveen Seshadri

End-user value of rich storage



How does WinFS/LH add value?

- Because I can organize data better?
 - I really want to relate each of my photos to a calendar entry!
- Because I can find the data on my desktop?
 - I really want to find documents from all people who I'm meeting with tomorrow!
- WinFS provides a rich data model, but....
 - The real problem is **information overload**.
 - I wish I could tell the computer what to do instead of doing it myself

Automation with customization is the killer app for WinFS

By the end-user

Over all data

In every application

Outline of the talk

- What can the end-user do? -- Demos
 - Querying of data – through logical data views
 - Organization of data – through automation rules
 - Customization of applications
- What are the elements of the rules platform?
 - End-user programming model
 - Common UX
 - Data model integration with the WinFS schema
 - API integration with the WinFS API
 - Execution using the WinFS query processor

End-User Programming Model

ON input IF condition THEN results

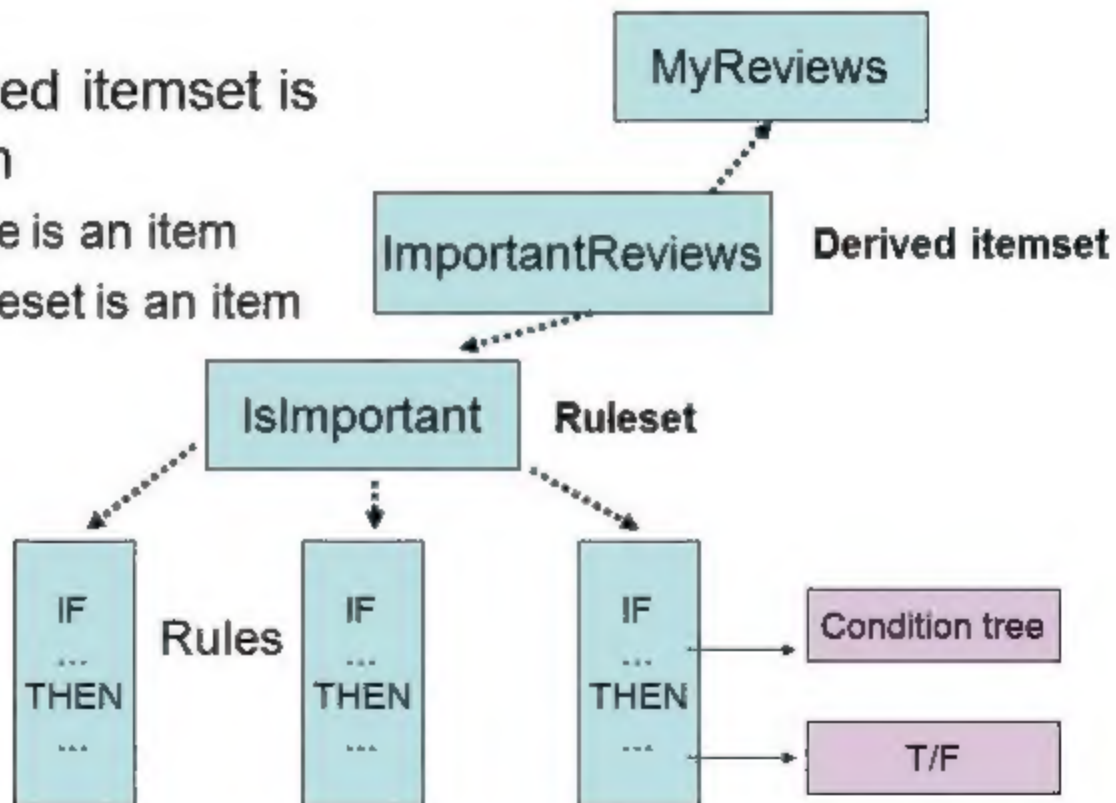
- Rule
 - Every rule has a signature defining input and result types
 - Input is a schematized WinFS item *
 - Condition is a Boolean expression
 - Allowed base conditions are driven by item schema
 - Rich set of conditions allowed
- RuleSet
 - A set of rules to apply together
 - Conflict resolution (eg. based on priority order) across rule results
 - RuleSet is the “deployable” unit of logic
- A RuleSet is a declarative function

End-user Queries as rules

- An end-user query is:
 - A ruleset whose result type is Boolean (t/f)
 - An input item scope
- An end-user query defines a “derived itemset”
 - It acts like a read-only WinFS collection
- A derived itemset is stored as an item
 - It is persisted as a structured object
 - It can be synchronized like any other item
 - It is managed, secured, backed-up, etc like any other item
 - The queries can themselves be queried

End-user Queries as Items

- A derived itemset is an item
 - A rule is an item
 - A ruleset is an item

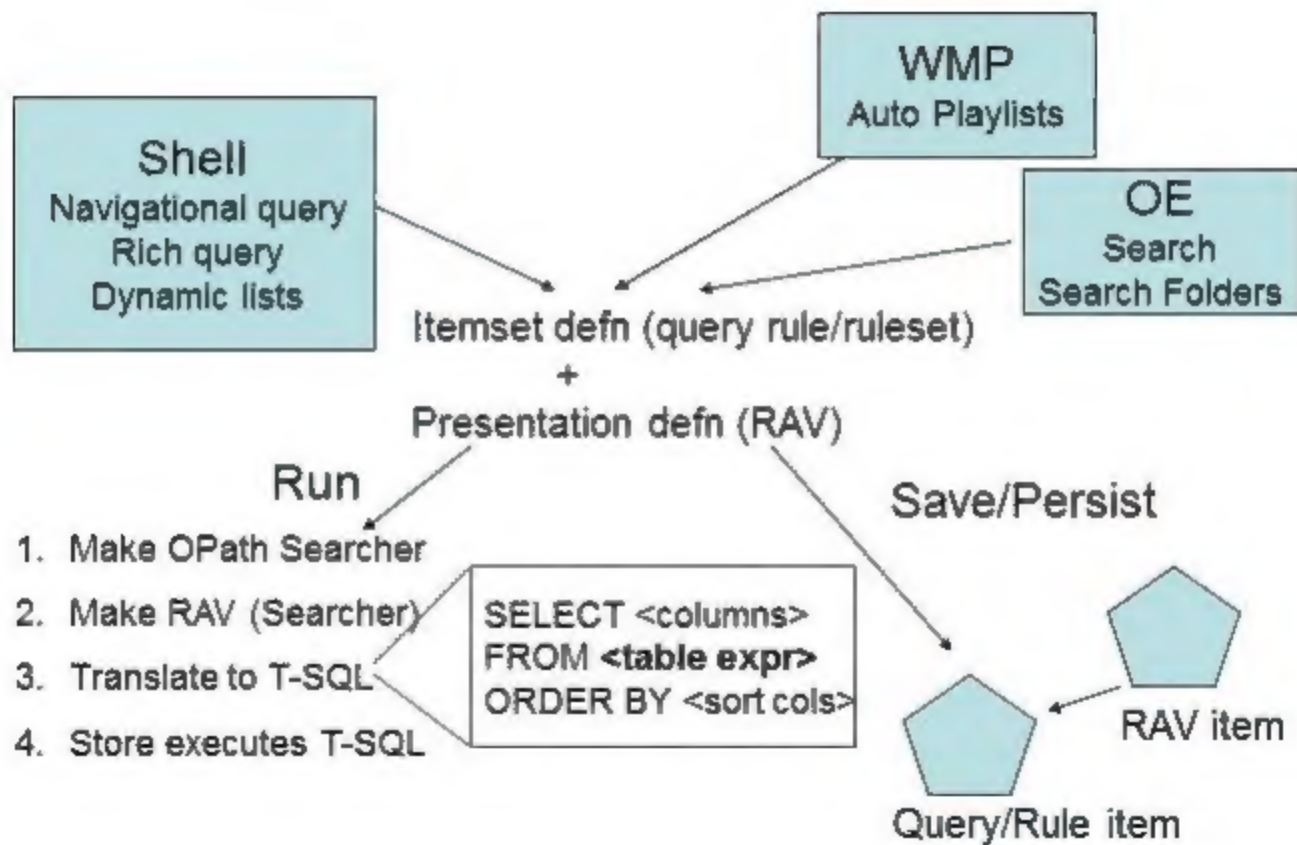


Composition & Sharing

- Composition:
 - A query algebra over itemsets
 - Scope composition: ImportantReviews can be the input scope for another derived itemset
 - Condition composition: "Find all ImportantReviews authored by any of MyFriends"
 - Richer composition
 - Rulesets as condition filters -- IsImportant
 - Virtual values – NextMeeting, HighestMailPriority
- Sharing:
 - Rules can be shared across rulesets
 - Rulesets can be shared across derived itemsets
 - Useful for corporate deployment scenarios

How does it work?

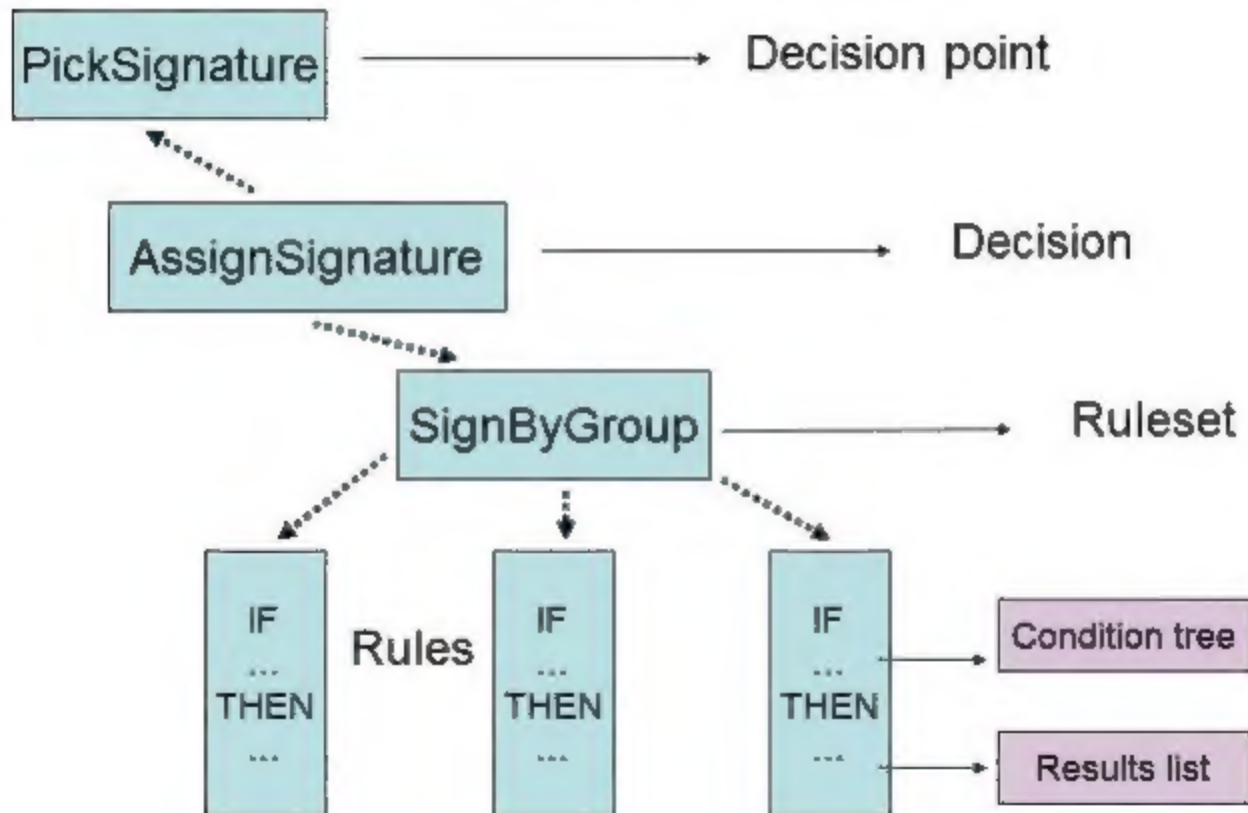
- Basic model (for queries and all other uses of rules)
 - Rules require data matching – joins, indexes
 - Use the SQL query processor as the real engine
 - Provide rule semantics in a translation layer
- Derived itemsets are accessed via OPath
 - No difference in developer experience



Decision Rules

- Rules whose results can be more general than true/false
- Applied at application interceptor “events”.
 - E.g. Pick a signature file for a mail sent with Outlook
- Invoked synchronously
 - “Decision point” WinFS item used to define the event
 - A ruleset captures the end-user decision logic
 - The rules platform maintains the association of ruleset to decision point
 - The application invokes the decision point, and then acts upon the decision

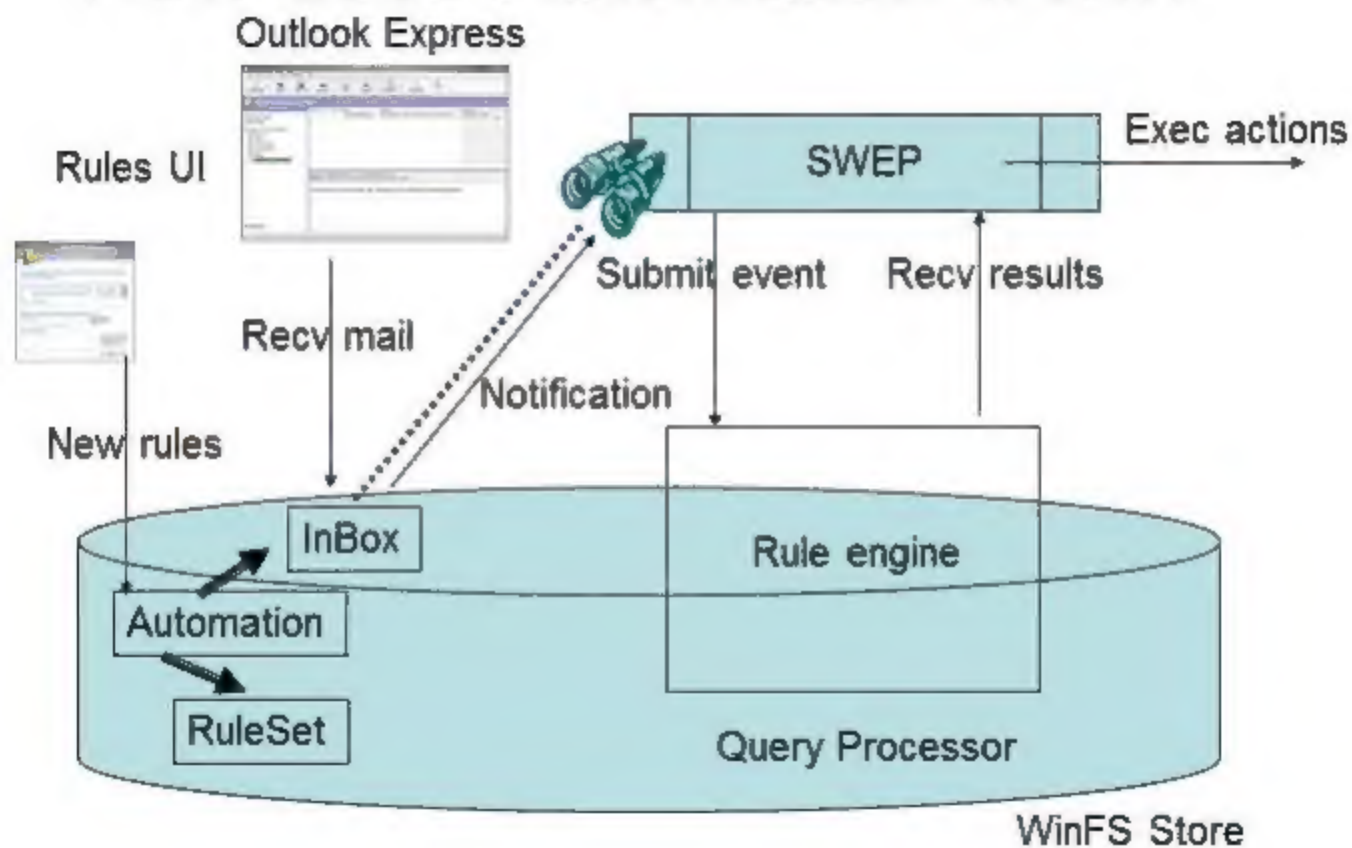
End-user Decisions as Items



Automation rules

- Decision points associated with the WinFS “active application”
 - The decisions are invoked asynchronously in response to data change events
 - Eg. an item got added to a particular folder
 - The results of the decisions correspond to methods/verbs/actions to invoke on the item
 - Eg: set Reviewer to Praveen
- Implemented by the “SWEF” component
 - Hosted in an app/service/Shell?

How does Automation work?



Questions?